Project Description

Network Monitoring for Performance Analysis and for Enabling Network-Aware Applications

Proposal for the Office of Science Financial Assistance Program 99-08

Next Generation Internet Research in Basic Technologies

Brian L. Tierney

Lawrence Berkeley National Laboratory

Joseph B. Evans and Victor S. Frost

University of Kansas

Abstract

Emerging NGI applications will push the limits of available network bandwidth. There are two critical services required to guarantee maximum efficient use of the network resources. The first is a system for monitoring the performance of each component in the system, enabling detailed performance analysis of the complete end-to-end system. The second is a system for monitoring current network characteristics, and providing this information to network-aware applications, which can effectively adapt to the current network conditions.

These capabilities require a very similar set of services. Both require an adaptive monitoring infrastructure, a monitor data publishing mechanism, and monitor data analysis tools. We propose to develop a "grid" service that will provide both of these capabilities. The overall goal of this work is to address these issues in order to provide manageability, reliability, and adaptability for high performance applications running over wide-area networks.

1.0 Introduction and Overview

The next generation of high-speed networks will allow DOE scientists unprecedented levels of collaboration. Large data archives will be easily accessed from anywhere on the network. However, diagnosing performance problems in high-speed wide-area distributed systems is difficult because the components are geographically and administratively dispersed, and problems in one element of the system may manifest itself elsewhere in the network. Problems may be transient, and may be due to activity in the infrastructure. Also, a large volume of monitoring data may be needed for diagnosis and the type of monitoring data and its analysis depends on the nature of the problem, and the necessary monitoring data may not be available when it is needed.

In addition to the ability to locate performance problems, in order to efficiently use NGI networks this new class of applications will need to be "network-aware"[10]. Network-aware applications attempt to adjust their resource demands in response to changes in resource availability. Emerging QoS services will allow the application to participate in resource management, so that network resources are applied in a way that is most effective for the application. Network-aware applications will require a general-purpose service that provides information about the past, current, and future state of all the network links that it wishes to use.

We propose to develop and deploy a NGI service that addresses both of these issues. We are calling this service ENABLE (Enhancing of Network-aware Applications and BottLeneck Elimination). This service will include monitoring tools, visualization tools, archival tools, problem detection tools, and monitoring data summary and retrieval tools. The monitoring tools will be capable of monitoring the entire end-to-end system, and will include tools for monitoring network components (switches, routers, and links), system components (hosts, disks, etc.), and applications. The results of the monitoring will then be published in directory services via the Lightweight Directory Access Protocol (LDAP), allowing network-aware applications to obtain the information needed to adapt to current conditions.

Currently a variety of ad hoc techniques are used for performance analysis of distributed systems. The system we propose will provide a single, comprehensive, easy to use system to address this need. These tools will make it much easier to locate problems or bottlenecks in NGI "computational grid" applications [13][15], and we believe that ENABLE will be one of several important "grid services" in the NGI environment.

1.1 Important New Capabilities

System monitoring is critical for diagnosing problems and doing performance tuning of distributed systems. We propose to develop tools that will provide accurate, adaptive, and on-demand monitoring of all aspects of distributed systems. The approach involves real-time analysis methods for anomaly identification and response, problem-characterization through real-time correlation techniques, and codification of expert knowledge for problem analysis and prediction. Our approach is novel in that we are combining network-level (via SNMP), and application-level (TCP, http, etc.) monitoring to provide a complete, real-time, picture of the state of the system and network. We also propose to archive all monitoring data, which will allow us to answer the question, "How did we get to this state?", and to make this monitoring data available to network-aware applications.

In addition to helping diagnose performance problems, this type of network monitoring will provide the necessary information to enable network-aware applications. For example LBNL has demonstrated large increases in network throughput in a network aware client/server application that uses network link throughput and delay information to set TCP send and receive buffers to the optimal size of a given link [16]. This application currently uses statically defined maximum link throughput information, along with dynamically determined delay information, to compute the optimal buffer sizes. The ENABLE service will make this much simpler by providing the client and server with the optimal buffer size to use for a given link.

Network-aware applications must be able to obtain information about resource availability, in particular, the network's capabilities and status. Unfortunately, network architectures differ significantly in their ability to provide such information to an application. In order to avoid

dependencies on idiosyncrasies of network architectures, applications require a system-independent interface to inquire that state of the network. This allows the development of portable applications that can adapt on any network that can provide adequate information. The ENABLE service will be able to provide this interface to network-aware applications.

Many of the components of computational grid applications are likely to be "precious" resources. For example, supercomputers, scientific instruments, and high-speed network links are all very expensive, and there is a desire to use these resources as efficiently as possible. High-speed networks are greatly increasing the accessibility of these resources, but they also increase the chance of "wasting" the resource. For example, if a large batch job on a supercomputer must wait for data from a remote archive, chances are CPU cycles will be wasted. Another example is a researcher may have one of the Advanced Light Source (ALS) beam lines reserved for a remote controlled experiment, but the network becomes congested during the time of the reservation, making remote control impossible. The ENABLE service can be used to provide support to resource reservation systems such as Globus to help determine which resources must be reserved in advance so that resources are not wasted.

Multimedia applications might make use of the ENABLE system to select the appropriate service levels in an incremental manner. For example, ENABLE might detect congestion problems during initial use of the network by an application. Should this application be sufficiently privileged, it might then request specific resource reservations from the quality of service infrastructure available from the network. This might enable the use of lower-cost best effort services when the needed performance is available, and higher cost options such as private networks with resource reservations only when absolutely necessary for a particular application.

1.2 NGI Expertise

Research groups at both the University of Kansas (KU) and at Lawrence Berkeley National Laboratory (LBNL) have a long history of working with NGI-like networks. LBNL has been involved with several high-speed network projects, starting in 1993 with BAGNET (Bay Area Gigabit Network) [4] and BLANCA [5]. Since then, LBNL has been involved with the DARPA MAGIC testbed [17], NTON [25], CAIRN [6], and SuperNet [32]. LBNL has worked on a variety of systems for handling scientific data over these networks, included earth imagery, high energy nuclear physics data, medical imaging data, and high-resolution high frame rate digital video data. LBNL's focus has been to tune distributed applications for high performance in a high-speed WAN environment. KU has been involved with several high-performance DARPA network testbeds such as MAGIC, the AAI [1], and CAIRN, as well as proprietary testbeds such as the Sprint SPARTAN [31] effort. KU's focus has been on network performance issues, and has included a strong emphasis on host and network testing and system monitoring.

We believe that this combination of expertise: distributed application expertise at LBNL and high-speed networking expertise at KU, is exactly what is required to make this sort of service successful. Most network monitoring tools are developed without close cooperation of application developers, and most application developers are unaware of what network monitoring information is available, or even useful. This combination of expertise will allow the ENABLE service to be valuable to both network administrators and distributed application developers and users.

2.0 Background and Motivation

2.1 Current Problems

Developers of high-speed network-based distributed systems often observe performance problems such as unexpectedly low network throughput or high latency. The reasons for the poor performance can be manifold and are frequently not obvious. It is often difficult to track down performance problems because of the complex interaction between the many distributed system components, and the fact that performance problems in one place may be most apparent somewhere else. Bottlenecks can occur in any of the components along the paths through which the data flow: the applications, the operating systems, the device drivers, the network adapters on any of the sending or receiving hosts, and/or in network components such as switches and routers. Sometimes bottlenecks involve interactions among several components or the interplay of protocol parameters at different points in the system, and sometimes, of course, they are due to unrelated network activity impacting the operation of the distributed system.

While post-hoc diagnosis of performance problems is valuable for systemic problems, for operational problems users will have already suffered through a period of degraded performance. The ability to recognize operational problems would enable elements of the distributed system to use this information to adapt to operational conditions, minimizing the impact on users.

The first goal of this performance characterization work is to produce high-speed components that can be used as building blocks for high-performance applications, rather than having to “tune” the applications top-to-bottom as is all too common today. This method can also provide an information source for applications that can adapt to component congestion problems.

The second goal is to help provide an infrastructure for enabling network-aware applications and middleware. Emerging QoS implementations will allow network-aware applications adjust to changes by modifying their resource demands. Services with a QoS assurance are more expensive than best-effort services, so applications may prefer to adjust rather than pay a higher price. Moreover, an application purely based on QoS assurance must be able to determine its resource demands in advance and limit resource usage to those provided by the QoS service, which is a significant restriction on flexibility. The ENABLE service we propose will aid the application in deciding which QoS it really needs or is willing to pay for.

2.2 Preliminary Studies

LBNL has been developing a methodology that enables real-time diagnosis of performance problems in complex high-performance distributed systems, which we call the NetLogger Toolkit [21]. NetLogger includes tools for generating precision event logs that can be used to provide detailed end-to-end application and system level monitoring, and tools for visualizing log data to view the state of the distributed system in real time. NetLogger has proven to be invaluable for diagnosing problems in networks and in distributed systems code [30]. This approach is novel in that it combines network, host, and application-level monitoring, providing a complete view of the entire system. NetLogger monitoring allows us to identify hardware and software problems, and to react dynamically to changes in the system.

LBNL has also been developing software agents to provide a wide range of management and

monitoring functions for distributed resources. We call this system Java Agents for Monitoring and Management (JAMM) [14]. The ability to monitor and manage distributed computing components is critical for enabling high performance data intensive computing. As distributed systems become bigger and more complex, there are more pieces to monitor and manage. Our approach to this problem is to use a collection of software agents that can perform various administrative tasks. Agents can securely start any monitoring program on any host, and manage the output of any monitoring data. For example, we are currently using the JAMM agents to trigger network throughput and latency monitoring tools, and publish their results into an LDAP database for easy access by network-aware applications or servers, such as the DPSS.

At the University of Kansas we have developed NetSpec [22], a tool designed to simplify the process of doing network testing, as opposed to doing point to point testing. NetSpec provides a fairly generic framework that allows a user to control multiple processes across multiple hosts from a central point of control. It consists of daemons that implement traffic sources/sinks and various passive measurement tools to provide convenient and sophisticated support for experiments testing the function and performance of networks. NetSpec provides a far wider range of test types and scenarios than prior methods (e.g. ttcp or netperf), because accurately characterizing network behavior and performance requires subtler testing than what is provided by a "full blast" data stream from point A to point B. NetSpec uses a scripting language that allows the user to define multiple traffic flows from/to multiple computers. This allows an automatic and reproducible test to be performed. NetSpec consists of several daemon types that are started and controlled by the main NetSpec daemon.

The University of Kansas has been collecting measurements from the AAI and MAGIC testbeds since late 1995. This experience led us to develop NetArchive. The NetArchive architecture includes a configuration database, time series database, traffic and connectivity information collectors, and various plot and information summary utilities. This toolset has been released to the Internet community for general use. Included among the types of information collected are throughput measurements based on switch cell and router packet counts, as well as connectivity and round-trip time information based on ping. To date, the network measurements have been used to characterize network traffic and to identify network performance bottlenecks.

In addition to building on previous work at LBNL and KU, we plan to use the Globus MDS service [8] to publish important ENABLE data. We also hope to use the NPACI Network Weather Service (NWS) [26] for network prediction after it becomes available for Globus.

The ENABLE service is somewhat similar to the Globus [12] network performance tool, GloPerf, which supports dynamic measurement and publication of network and bandwidth information. However ENABLE will provide a lot more information than is currently available by GloPerf. We plan to integrate ENABLE with GloPerf as part of this proposal.

2.3 Related Work

There currently are many network monitoring projects going on. For example, there are over 100 measurement and monitoring tools and projects listed on the Cooperative Association of Internet Data Analysis (CAIDA) Web page (http://www.caida.org/Tools/taxonomy.html). However, only a few of these tools address application performance, and those tools address only specific applications. Our approach is valid for any application. We plan to build a framework that is general

purpose enough that several of the CAIDA tools could be incorporated by modifying them to use the NetLogger log format and integrating them into our monitoring infrastructure.

In addition to the CAIDA tools, several studies in network traffic characterization have been done [3, 27, 28]. However none have directly correlated the traffic with instrumented applications, and none have focused on network-aware applications with very large bandwidth requirements.

2.4 Generic Tools to be Developed

As part of this proposed project, we will develop a number of network monitoring tools and services. These will include:

- Tools for network, system, and application monitoring
- Tools for management of monitoring data
- Tools for archiving monitoring data
- Tools for performance analysis and anomaly detection
- Tools for queries on historical monitoring data
- Support for publishing important monitoring data and summaries of results
- Application API for common queries of published results
- Integration with QoS systems
- Integration with existing Globus services
- Integration with future Globus services, such as the network resource brokering service
- Security mechanisms for the collection, distribution, and access of monitoring data

This will lead to a general-purpose grid service that will enable high performance NGI applications to effectively and efficiently use NGI networks. For example, the ENABLE service can be used by services such as the *High-Performance Data Transfer Service* proposed in the NCAR *Earth System Grid* proposal (currently submitted to Program Notice 99-09) to allow users (or applications) to express relatively high-level specifications of network requirements. The *High-Performance Data Transfer Service* will help hide the complicated details from the users and it will be responsible for locating, reserving, and configuring appropriate resources so as to ensure required end-to-end quality of service.

We will deploy the monitoring tools on ESnet, NTON, CAIRN, SuperNet, or other NGI testbeds doing NGI application development research. This will provide support for detailed performance analysis of the networks and applications in these testbeds, which are a prototype of the Internet environment of the future.

This work will build on the monitoring agent framework work at LBNL that is currently is being deployed in the China Clipper project [7]. China Clipper is the first DOE NGI application testbed, and is building a data management environment for High Energy Nuclear Physics data with ANL, LBNL, and SLAC.

A large number of NGI-related projects will benefit from this service. These include the China Clipper project, Globus [12], and DARPA's MATISSE project [18]. It is also highly relevant to the SSI and ASCI projects, and other projects in the NSF PACIs, DARPA, and the NASA Information Power Grid program [20]. More information on how ENABLE might be used in other projects is in section 7 below.

3.0 Preliminary Work

In our previous work we developed the basic methodology for performing diagnosis of performance problems in complex high-performance distributed systems. This methodology, called NetLogger[1] [27], includes tools for generating precision event logs that can be used to provide detailed end-to-end application, network, and system level monitoring. It also includes tools for visualizing the log data and real-time state of the distributed system. NetLogger has proven to be invaluable for diagnosing problems in networks and in distributed systems code. Other previous work also includes a variety of network measurement, monitoring, and archiving tools. The NetSpec [22] toolset allows complex networks to be actively probed and measured for connectivity, throughput, and delay using a variety of test daemons on network nodes. The NetArchive system [23], portions of which have been active on the AAI and MAGIC testbeds for several years, includes traffic measurement collection capability based on SNMP, ping, and other sampling methods. NetArchive also includes an SQL configuration tracking database, archival time series databases based on the NetLogger log format, and a variety of plotting and executive summarization tools for web display. We describe each of these in detail below.

3.1 NetLogger

NetLogger consists of *event logs* that represent the raw information about system performance, and the *NetLogger Toolkit* that generates and manipulates the logs.

To analyze the performance of a wide-area distributed system, it is important to log as much information about the state of the system as possible. NetLogger *Event logs* contain high-resolution, synchronized timestamps taken before and after *events* (activities of interest). Events may include application-, operating system-, or network-level activities. Monitoring of operating system and network conditions is an important complement to application-level monitoring. To aid in the processing of the potentially huge amount of log data that can be generated from this type of logging, NetLogger logs all data using a common format: the IETF draft standard Universal Logger Message format (ULM) [33]. To allow comparison of log data from multiple hosts, the clocks of all hosts participating in the distributed system being synchronized via the NTP protocol. To use NetLogger to analyze a distributed application, the application must be instrumented to log the time at which data is requested and received, and record any local processing time.

The NetLogger Toolkit consists of three components: a library of routines to simplify the generation of application-level event logs (for C, C++, and Java™), a set of modified operating system utilities, a set of tools for managing and filtering the log files, and a set of tools to visualize and to analyze the log files. The library contains routines to open, write, and close log files. Events can be written to a local file, the syslog daemon, or a given TCP port on a given network host.

In NetLogger terminology, the logical path of an object (a particular datum or process flow) through the system is called its lifeline. This lifeline is the temporal trace of an object through the distributed system. NetLogger builds lifelines by combining specified events from a given set of processes, and represents them as lines on a graph. NetLogger plots time (i.e., the timestamp from the event log)

[1] The name NetLogger is somewhat misleading, as it is designed to log much more than just network information. A more descriptive name would be “comprehensive distributed system component Logger”, but NetLogger is shorter and easier to remember.

against a set of events. For example, in a client-server distributed system, each request-response transaction might be represented as a lifeline; the events on the lifeline might include the request's dispatch from the client, its arrival at the server, the commencement of server processing of the request, the dispatch of the response from the server to the client, and the arrival of the response at the client.

Exploratory, interactive analysis of the log data—especially analysis of the graphical representations of individual, exceptional events—has proven to be the most important means of identifying the causes of specific behavior. In particular, the ability to distinguish, manipulate, and analyze lifelines is critical to isolating the locations of (and thereby the reasons for) unexpected behavior. LBNL has developed a tool called *nlv* (NetLogger Visualization) for interactively viewing the NetLogger event files. *nlv* can display several types of NetLogger events. The user can combine multiple different sequences of events, servers, and graph types (lifeline, load-line, or point) on a single graph; the display can be modified to show an arbitrary subset of these elements.

Monitoring of operating system and network activities complements application-level monitoring. Characterizing a distributed system's performance requires distinguishing between the "application" and its supporting infrastructure. NetLogger monitors and logs operating system- and network-level events using versions of the Unix utilities *netstat* and *vmstat* that have been modified to support NetLogger. *netstat* reports on the contents of various network-related data structures. *vmstat* reports statistics on virtual memory, disk, and CPU activity.

LBNL has been using the NetLogger tools for performance analysis for the DOE China Clipper project [7]. The goal of the China Clipper project is to provide high-speed remote access to High Energy Nuclear Physics (HENP) data over NGI-like networks. Using the STAF application (a HENP data analysis package), we achieved remote I/O of 57 Mbytes/sec from LBNL to SLAC over NTON. This is the equivalent of 4.5 Terabytes/day. The NTON link to SLAC was an end-to-end OC-12 (622 Mbits/sec) ATM. This is the rate for application data from parallel disks to client memory, and this rate was achieved using a 4 server Distributed Parallel Storage System (DPSS) at LBNL, and the client was running on an 8 CPU Sun server at SLAC.

Similar experiments between LBNL and ANL over Esnet (2000 km), which is an IP routed OC-12 network, resulted in an end-to-end throughput of 35 MBytes/second. NetLogger was used to help adjust several buffer and pipeline size, and to determine that the current bottleneck is the client host, a two CPU Sun workstation.

The NetLogger toolkit has been essential to obtaining these rates in the China Clipper project. A lot of client and server tuning were required, and NetLogger allowed us to locate where problems were and whether our changes helped performance.

NetLogger was released to the community in mid 1998, and is being used by many organizations, including ANL, KU, NASA, SRI, and several groups at LBNL.

3.2 Agents for Event Logging Management

Management of monitoring programs and event logs for many clients connected to many distributed server components on many hosts is quite difficult. Our approach to this problem is to use a collection of software agents to provide structured access to current and historical information.

LBNL has developed a software agent architecture for distributed system monitoring and management, which is called JAMM (Java Agents for Monitoring and Management) [14]. The agents, whose implementation is based on Java and RMI, can be used to launch a wide range of system and network monitoring tools, extract their results, and publish them into an LDAP database.

These agents can be used to run any system monitoring utility. For example, we use the agents to run *netperf* [24] and *ping* for network monitoring, and to *vmstat* and *uptime* for host monitoring, and *xntpdc* for host clock synchronization monitoring. These results are uploaded to an LDAP database at regular intervals, typically every minute, for easy access by any process in the system. We run these agents on every host in a distributed system, including the client host, so that we can learn about the network path between the client and any server.

The agent architecture is a crucial component for a monitoring system because without it management of the immense volume of log event data that can be generated would be infeasible. In addition, the agent architecture is sufficiently general to be useful for a wide variety of distributed system management tasks. We are also using these agents in other projects to make sure servers are up, and to help with load balancing and fault tolerance. We expect to develop new uses for this agent architecture as we gain experience using it with different types of applications.

3.3 NetSpec

NetSpec is a network-level end-to-end performance evaluation tool developed by researchers in the University of Kansas, to help in the collection of results delivered by performance experiments on the ACTS Satellite ATM Internetwork (AAI) project. The NetSpec system provides support for large scale data communication network performance tests with a variety of traffic source types and modes. This software tool provides a simple block structured language for specifying experimental parameters and support for controlling performance experiments containing an arbitrary number of connections across a LAN or WAN.

NetSpec exhibits many features that are not supported by the most often performance tools used today (ttcp, Netperf), like parallel and serial multiple connections, a range of emulated traffic types (FTP, HTTP, MPEG, etc.) on the higher levels, the most widely transport protocols used today, (TCP and UDP), three different traffic modes, scalability, and the ability to collect system level information from the communicating systems as well as intermediate network nodes.

The basic NetSpec architecture consists of several pieces. The controller is a process that supports the user interface, which is currently a file containing a description of an experiment using a simple block structured language in which the connection is the basic unit for an experiment and via the control daemon controls the daemons implementing the test. For every connection in the experiment, the corresponding test daemons are created. These test daemons concentrate on performing the traffic related tasks (send or receive data transferred across the connection). Each daemon is responsible for its own report generation after experiment execution is complete, and measurement daemons concentrate on collecting data as accurately as possible, without having to worry about performing traffic functions. The output report is delivered to the controller via the control daemon for viewing by the user. The communication between the controller and the daemons is achieved using an ASCII based language, which enhances portability and extensibility.

NetSpec supports three basic traffic modes: full blast mode, burst mode, and queued burst mode. These basic traffic types provide the basis for a variety of system evaluation and debugging approaches.

NetSpec has the potential to emulate FTP, telnet, VBR video traffic (MPEG, video-teleconferencing), CBR voice traffic, and HTTP (World Wide Web traffic) on the application layer. This feature makes NetSpec a unique network performance evaluation tool with the ability to test system performance under traffic conditions that would be experienced in operation.

3.4 NetArchive

The Network Monitor Archive is a database of network measurements collected from various testbeds. Currently sites on the AAI and MAGIC are monitored. Included among the types of information collected are throughput measurements based on ATM switch cell and IP router packet counts, as well as connectivity and round-trip time information based on ping. The Network Monitor Archive architecture includes the Configuration Database, Time Series Database, traffic and connectivity information Collectors, and various plot and information summary utilities.

The Configuration Database contains information on the equipment to be monitored, and the specifics of the physical and logical interfaces that are being observed. Included are timestamps indicating the beginning and end times of the measurements for that entity. This allows queries to be formulated regarding the active devices within certain time periods. Other attributes include type of equipment, as well as special display directives.

The time series database contains the actual measurements collected by the system. These measurements are stored in NetLogger format for easy integration with other tools. The measurements are stored using Unix directories and files for efficient retrieval by the display tools. Compression of the measurement files is optionally enabled.

The Collector gathers traffic and connectivity measurements via a variety of tools, such as SNMP queries and ping probes. The Collector retrieves information from the monitored devices based on the entities specified in the Configuration Database, and stores the data in the Time Series Database.

A variety of display tools are included, such as a thumbnail generator for rapid perusal of commonly monitored entities, a more flexible archive plotter for complex queries, a real-time plotter so that currently active streams can be monitored during experiments, and a summary generator so that high level information on usage and connectivity over time periods can be displayed.

Note that various functions can be distributed to different physical machines in the network(s).

4.0 Technical Approach

The event analysis methodology proposed here will integrate and build on previous work from LBNL and KU to enable the real-time diagnosis and prediction of performance problems in wide-area networks. A general monitoring architecture will be constructed, and experiments will be conducted with several high-bandwidth applications to demonstrate its utility and to explore its limits.

The system we are proposing to develop will address end-to-end performance issues, with a particular emphasis on high speed, wide area systems. The architecture must include the following features:

- Support for measurements on end systems and networks
- Support for measurements spanning multiple networks
- Support for correlation of events such as packet loss and application delays
- Support for measurements with various time granularities
- Robustness to network failures and measurement probe errors
- Scalability to many distributed applications
- Simplicity of use by application users with no specialized training
- Visualization of the measurements and correlations in a form usable by application users with no specialized training
- Aids for identifying and debugging anomalous conditions

The key components to enabling real-time diagnosis of wide-area network performance problems include a complete set of network and application level monitoring tools, and the ability to trigger what gets monitored based on the current network performance and configuration. It is also important to have the ability to archive all monitoring data, allowing one to go back and see what events lead up to the performance problem.

Several questions will need to be explored to achieve real-time diagnosis of performance problems. These include:

- What events should be monitored?
- How often should they be monitored?
- Is active or passive monitoring more useful in a given situation?
- How much does active monitoring effect the network and applications on the network?

A management agent framework will make it possible to explore these questions. Software agents will be used to control what gets monitored, and how often it is monitored. The agents can be used to increase or decrease the level of monitoring based on current network performance, or based on what applications are currently being run.

The proposed system includes tools for measurement of the end systems and networks involved in end-to-end transactions, tools for collecting and archiving the results of the transactions, tools for analyzing and correlating measurements as well as detecting anomalous conditions, and tools for visualizing the outcomes. Our proposed architecture will be implemented using existing technology (both standard tools and those developed by the investigators) wherever possible, and tools developed under this proposed work where necessary. We will discuss each major element of the architecture in turn.

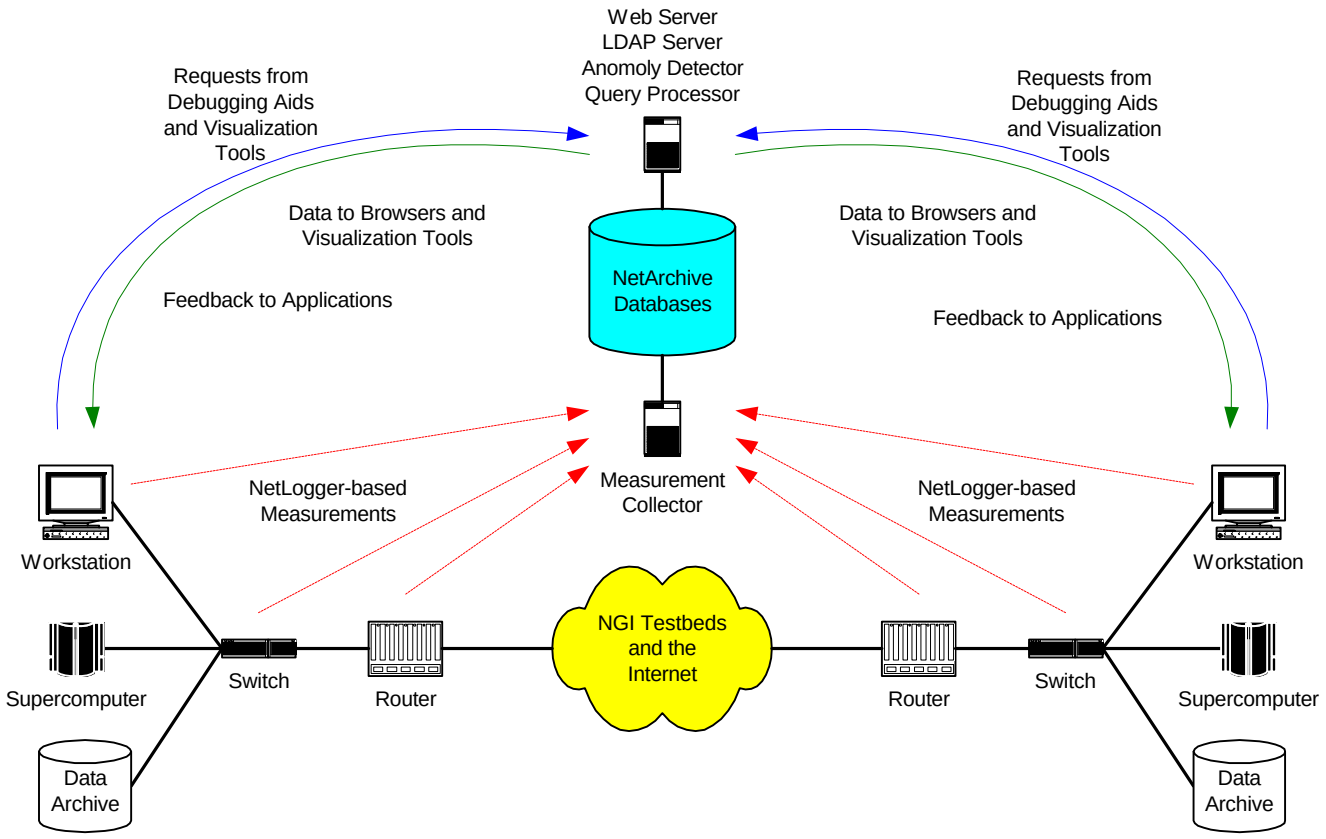

Figure 1: ENABLE Architecture and Operating Environment

4.1 Measurement tools

In this context, measurements will include both host-based data and network data. These include the various tools such as NetSpec developed and integrated into an agent framework in earlier work. Other tools such as SNMP-based collectors will be integrated into the agent framework as part of this effort in order to enable manageable end-to-end measurements.

A measurement library will be developed and distributed in order to provide application writers with easy access to tools for network wide debugging. This library will have a simple API that will permit application writers to enable logging of selected events to specified logging archive. Specific common applications such as ftp and web tools will be extended to include measurement capability.

All of these tools will use the NetLogger approach for logging measurements which will be archived using the NetArchive system.

4.2 Archive management tools

The ability to understand complex application and network interactions is supported by the archive system. This will be based on the NetArchive system developed at KU. This system uses an SQL relational database to record various attributes about sessions that are being measured, and contains pointers to NetLogger event log files. In the proposed work, we will extend the NetArchive system to support larger database sizes and more sophisticated retrieval of information, and integrate the

system into the Globus framework.

4.3 Visualization tools

Visualization tools are necessary to understand the interactions between applications, host performance, and network conditions. In this work, we will develop tools for visualization of application interactions with the network such as reads, writes, and parameter modifications, host protocol events such as retransmissions, and network events such as congestion events. These tools allow users to observe and understand the significant events that are affecting performance.

The infrastructure for this visualization is inherent in NetLogger and the NetArchive systems. Time correlation between events is made possible by the event logs, and the visualization tools need to express this in useful ways. Preliminary work in this area by LBNL provides the ability to view certain NetLogger event on X-Windows displays. The proposed work will extend that to provide additional flexibility in terms of events that may be viewed, and the visualization technology (such as Java applets). In addition to visualization of events, tools will be provided to correlate the events with application and host parameters, and current network topology. The former can be supported using the existing NetLogger approach. The later will involve collection of additional data such as routing through tools similar to traceroute. These will be integrated with the anomaly detection tools to help pinpoint particular problem areas.

4.4 Anomaly detection tools

The anomaly detection tools will detect conditions in the applications, hosts, and networks, which lead to poor behavior.

These tools will be based on two approaches: (1) direct observation of parameters and behavior, (2) correlation of past network patterns with current observations. The first approach might involve, for example, the observation of TCP window sizes from traffic samples obtained via the tcpdump tool, and identifying windows that are not open sufficiently for the measured round-trip time. The second approach might be used when many similar experiments are being performed, so that correlation is enabled. For example, repeated file transfers that exhibit poor performance during certain times of the day and good performance during others on the same machines might be explained by correlation with switch or router congestion conditions during certain parts of the day.

The effectiveness of these two approaches will be evaluated in the course of this research.

4.5 Deployment and Testing

An important component of this effort is the deployment and use of the system, not merely design and implementation. These tools will be used in DoE networks and various NGI testbed networks to improve performance in various settings. These technologies will be deployed in DoE networks for testing with applications which transfer high data volumes, and for which peak performance is critical. NGI testbeds such as NTON [25] and ATDNet [2] will also be used as proving grounds for these tools. The deployment will involve configuration of network accessible hosts as measurement and archive sites, as well as instrumentation of selected applications for measurement.

4.6 Grid Service Application API

This monitoring infrastructure can be used to provide network-aware applications with the information required to more efficiently utilize the network. We will determine what information is the useful to applications and develop a client API for accessing this information. This API will includes calls to:

- Recommend the optimal TCP buffer sizes to use
- Report on current throughput and latency information for a given network link
- Recommend which protocol to use
- Recommend which compression level to use
- Recommend if QoS is required, or if best effort service is likely to be good enough
- Report future network link prediction, based on the Network Weather Service information

In addition to the calls listed above, we will follow other research in this area, such as the Darwin Project at CMU [9], to create a general-purpose service for network-aware applications.

4.7 Standardization and Distribution

As part of this effort, we will actively pursue the standardization of the relevant portions of this work through the IETF, and the distribution of the tools developed in this work to DoE users and application developers.

5.0 Work Plan

This effort will involve the University of Kansas (KU) and Lawrence Berkeley National Laboratory (LBNL). Each task will have a lead organization, with contributions from other organizations on certain tasks. All work will be coordinated with regular conference calls, and regular integration and testing exercises. Each organization has submitted a separate proposal with a separate list of work items, and the two proposals should be considered together.

Lawrence Berkeley National Lab Work Items

Task 1 – Measurement Management Tools

An agent-based system will be developed for managing starting and stopping each of the monitoring tools, and controlling the frequency with which they are run. Tools will be developed to automatically trigger more monitoring when certain criteria are met, such as high traffic loads, high loss rates, or high certain applications are started.

Task 2 - Archival Management Tools

Tools for collecting, distributing, replicating, and filtering the log files will be developed. Monitoring data, the network topology and configuration at a given time will also be archived. All archived data will be accessible via a secure web server.

Task 3 - Visualization Tools

Tools for both real-time and historical analysis will be developed. Visualization tools to aid in event correlation will be developed as well. These tools will allow users to view the performance of the system prior to, during, and after use. They will allow network managers to study the performance of the network and under operational conditions and correlate with application usage. They will allow administrators to view the state of the network and resource usage through summarization tools. LBNL and KU will each work on a subset of these tools.

Task 4 - Integration with Globus

The ENABLE service with be integrated with GloPerf and other Globus services to become a standard "grid" service, and will be able to be used by any Globus client. This will hopefully involve working closely with researchers at Argonne National Lab (ANL) to integrate this work into their network resource broker.

Task 5 – Deployment

LBNL will work closely with KU to deploy these tools in NGI networks such as NTON, ESNet, MREN [19], CAIRN, and SuperNet. This will allow us to validate the utility of the tools immediately in a real network environment. We will use the archival features of these tools to support retrospective analysis of network performance and usage, as well as the more immediate applications. We will work with NGI application developers to instrument their applications with NetLogger monitoring, and to make these applications network-aware of other services such as QOS support that will be provided by the ENABLE service.

Task 6 – Application API

LBNL will work with KU to define and develop a network-aware application API to use the information acquired by the ENABLE service. This will include inquiry functions such as optimal TCP buffer size, network performance and latency, QoS availability, and so on.

University of Kansas Work Items

Task 1 – Network and Application Monitoring and Archiving Tools

These tasks will include the acquisition or development of tools for monitoring and archiving network and application measurements. SNMP-based tools will be used to collect measurements from IP routers, ATM switches, and hosts. Tools such as ping sweeps, and NetSpec (for TCP/UDP throughput and burst tests) will be used to monitor connectivity and link utilization. Preliminary work on such tools was done in DARPA and related efforts. All tools will use the NetLogger event log format. Other tools, such as those coordinated through CAIDA, will also be included as needed and as feasible. We will also instrument several applications with NetLogger, including the Apache web server, the Netscape web browser, and ftp clients and servers. The tool suite will include a relational database to support sophisticated queries regarding the measurements that have been collected.

Task 2 - Automatic Anomaly Detection Tools

Tools for the detection of anomalies in application and network performance will be explored. The types of anomalies of interest include application and network under-performance, application error conditions, and the more traditional network error behavior.

Task 3 - Visualization Tools

Tools for both real-time and historical analysis will be developed. Visualization tools to aid in event correlation will be developed as well. These tools will allow users to view the performance of the network and distributed system prior to, during, and after use. They will allow network managers to study the performance of the network and under operational conditions and correlate with application usage. They will allow administrators to view the state of the network and resource usage through summarization tools. LBNL and KU will each work on a subset of these tools.

Task 4 – Deployment

KU will work closely with LBNL to deploy these tools in NGI networks such as NTON, ESNet, MREN, CAIRN, and SuperNet. This will allow us to validate the utility of the tools immediately in a real network environment. We will use the archival features of these tools to support retrospective analysis of network performance and usage, as well as the more immediate applications. We will work with NGI application developers to instrument their applications with NetLogger monitoring, and to make these applications network-aware of other services such as QoS support that will be provided by the ENABLE service.

Task 5 – Application API

KU will work with LBNL to define and develop a network-aware application API to use the information acquired by the ENABLE service. This will include inquiry functions such as optimal TCP buffer size, network performance and latency, QoS availability, and so on.

6.0 Milestones

The proposed work is a three year effort. An emphasis will be placed on early development and deployment of prototypes so that system usage can be understood and improvements made in response to feedback from application developers.

Year 1 Milestones

- Identify first DOE NGI application to monitor (LBNL)
  - work with application developer to add NetLogger instrumentation to this application
  - deploy monitoring agents on all components used by this application
- Add NetLogger to httpd/netscape (KU)
- Integrate existing KU monitoring tools into agents (KU)
- Publish important monitoring results in Globus MCAT (LDAP) service for use by network aware applications (LBNL)
- Web-based queries on historical data (KU)
- Agent and log data security mechanism (LBNL)

- Publish NetLogger internet draft (LBNL)
- Demonstration of application with ENABLE (KU and LBNL)

Year 2 Milestones

- Automatic anomaly detection tools (KU)
- Scaling of NetArchive (KU)
- Globus integration (LBNL)
- Identify second DOE NGI application to monitor (LBNL)
  - work with application developer to add NetLogger instrumentation to this application
  - deploy monitoring agents on all components used by this application
- Package and release version 1.0 of ENABLE (KU and LBNL)

Year 3 Milestones

- Develop application API for most command ENABLE queries (KU and LBNL)
  - recommend the optimal TCP buffer size, report on current throughput and latency information for a given network link, recommend if QoS is required, etc.)
- Publish ENABLE client API (KU and LBNL)
- Integrate with QoS systems (IETF DiffServ: KU; Globus: LBL)
  - exploit feedback from ENABLE to select appropriate QoS levels
- Integrate with Globus resource brokering system (LBNL)
- Identify third DOE NGI application to monitor (LBNL)
  - work with application developer to add NetLogger instrumentation to this application
  - deploy monitoring agents on all components used by this application
- Package and release version 2.0 of ENABLE (KU and LBNL)

7.0 Linkages and Technology Transfer

The research proposed here complements other NGI proposals submitted by colleagues at LBNL, ANL, and other institutions. These proposals have been developed with the collective goal of defining and implementing an Integrated Grid Architecture for advance network applications. This work will provide a valuable grid service on which to enable performance analysis and to build network-aware applications. In particular, LBNL is involved in a number of NGI application proposals, including proposals for working with combustion, climate, and high energy physics data in at NGI environment. Additionally, we also plan to work closely with Ian Foster and Dan Reed, who have submitted a proposal titled “A Uniform Instrumentation, Event, and Adaptation Framework for Network-Aware Middleware and Advanced Network Applications”, to ensure our work is complimentary. For more information on how this relates to other NGI proposals, see Appendix A.

The proposed research is also highly relevant to the goals of ASCI DISCOM and Distance Corridor efforts and to other SSI application areas (e.g., Combustion) with a need for high-bandwidth transport and access to large amounts of data. All of these applications will greatly benefit from this proposed work.

We view the research described here as forming a key component of a larger activity designed to develop and apply an Integrated Grid Architecture. Hence, this work directly supports existing and proposed DOE research activities concerned with distributed computing (e.g., distance corridors,

climate modeling, materials science) and complements existing and proposed DOE research activities concerned with networking technology and middleware. In particular, this work will support projects concerned with resource management and quality of service, instrumentation, and network topology determination.

Several projects at KU would benefit from the ENABLE effort, such as the DARPA-sponsored Rapidly Deployable Radio Network (RDRN) project and several Active Networking efforts. These projects already use the NetSpec tool, and would benefit from the integration of experiments and measurements with sophisticated debugging utilities. Other projects, such as those investigating residential wireless access, are using NetArchive and would benefit from the ability to efficiently identify problems and bottlenecks

Software will be deployed, evaluated, and demonstrated on network testbeds, including by ESnet, NTON, VBNS, SuperNet, CAIRN, and MREN. Technology innovations resulting from this project will also be propagated to the larger community via the strong ongoing collaborative links that exist between the participants and other programs. In particular, we will work to get this services deployed by Globus, the NSF PACIs, DARPA, and the NASA Information Power Grid program.

KU and LBNL are also involved in a DARPA project to deploy tools for monitoring of the DARPA SuperNet testbed [31]. This experience will help us understand what are the issues involved in doing monitoring across multiple networks, and what needs to be monitored in other NGI testbeds.

8.0 Summary

The NGI will enable many new types of wide area distributed programming. It is our experience that distributed programs never perform as expected in a WAN environment. It will become increasing important to have a good monitoring and analysis system in place for use by this new class of applications.

We propose building an infrastructure to provide accurate, adaptive, and on-demand monitoring of all aspects of the distributed computing components, including the network. We will then use this monitoring data for real-time analysis and anomaly identification and response. This work will provide the tools to enable manageability, reliability, and adaptability for high performance applications running over NGI networks.

This monitoring infrastructure will also form the basis for a service to enable network-aware applications. It will make past, present, and future predictions of the network condition available to applications, which can then adapt their behavior to more efficiently use the network.

9.0 References

1. AAI: http://www.ittc.ukans.edu/aai/
2. ATDnet: http://www.atd.net/
3. M. Arlitt and C. Silliamson. "Internet Web Servers: Workload Characterization and Performance Implications," IEEE/ACM Transactions on Networking, 5(5):632-645, October 1997.
4. BAGNET: http://www-itg.lbl.gov/BAGNet.html
5. BLANCA: http://choices.cs.uiuc.edu/blanca/index.html
6. CAIRN: DARPA's Collaborative Advanced Research Network, http://www.cairn.net/
7. Clipper: The China Clipper Project, http://www-didc.lbl.gov/Clipper/
8. K. Czajkowski, I. Foster, C., Kesselman, N. Karonis, S. Martin, W. Smith, S. Tuecke. "A Resource Management Architecture for Metacomputing Systems", Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing,1998.
9. Darwin: Resource Management for Application-Aware Networks, http://www.cs.cmu.edu/afs/cs/project/cmcl/www/darwin/
10. DeWitt, T. Gross, T. Lowekamp, B. Miller, N. Steenkiste, P. Subhlok, J. Sutherland, D., "ReMoS: A Resource Monitoring System for Network-Aware Applications" Carnegie Mellon School of Computer Science, CMU-CS-97-194. http://www.cs.cmu.edu/afs/cs/project/cmcl/www/remulac/remos.html
11. DPSS: "The Distributed-Parallel Storage System", http://www-didc.lbl.gov/DPSS/
12. Globus: http://www.globus.org
13. Grid: *The Grid: Blueprint for a New Computing Infrastructure*, edited by Ian Foster and Carl Kesselman. Morgan Kaufmann, Pub. August 1998. ISBN 1-55860-475-8.
14. JAMM: http://www-didc.lbl.gov/JAMM/
15. W. E. Johnston. "Real-Time Widely Distributed Instrumentation Systems," In *The Grid: Blueprint for a New Computing Infrastructure*. Edited by Ian Foster and Carl Kesselman. Morgan Kaufmann, Pubs. August 1998.
16. J. Lee, B. Tierney, W.Johnston, B. Crowley, M. Holding, A Network-Aware Distributed Storage Cache for Data Intensive Environments, to be part of the proceedings of the 1999 IEEE Symposium on High Performance Distributed Computing.
17. MAGIC: "The MAGIC Gigabit Network.", http://www.magic.net/
18. MATISSE: MEMS application on the DARPA SuperNet, http://www.darpa.mil/ito/research/ngi/projects.html
19. MREN: Metropolitan Research and Education Network, http://www.mren.org/
20. IPG: NASA's Information Power Grid Project, http://www.nas.nasa.gov/IPG/
21. NetLogger; http://www-didc.lbl.gov/NetLogger/
22. "NetSpec: A Tool for Network Experimentation and Measurement", Information & Telecommunication Technology Center, University of Kansas, http://www.ittc.ukans.edu/netspec/
23. "Network Traffic Data Monitor Archive Architecture", Information & Telecommunication Technology Center, University of Kansas, http://netarchive.ittc.ukans.edu/network_archives.html
24. Netperf: http://www.netperf.org/

25. NTON: National Transparent Optical Network, http://www.ntonc.org/
26. NWS: Network Weather Service, http://nsw.npaci.edu/
27. V. Paxson and S. Floyd. "Wide-Area Traffic: The Failure of Poisson Modeling," IEEE/ACM Transactions on Networking, 3(3):226-244, June 1995.
28. V. Paxson, “End-to-End Internet Packet Dynamics”, SIGCOMM ’97, LBNL-40488.
29. B. Tierney, W. Johnston, J. Lee, “The NetLogger Methodology for High Performance Distributed Systems Performance Analysis”, proceedings of the 1998 IEEE Symposium on High Performance Distributed Computing.
30. B. Tierney, W. Johnston, J. Lee, G. Hoo, “Performance Analysis in High-Speed Wide Area ATM Networks: Top-to-bottom end-to-end Monitoring,” IEEE Networking, May 1996. http://www-itg.lbl.gov/DPSS/ papers.html
31. SPARTAN: http://www.spartan.net/
32. SuperNet: DARPA SuperNet Testbed, http://www.ngi-supernet.org/
33. ULM: Universal Logger Message Format, http://www.hsc.fr/veille/ulm/

Appendix A

An Integrated Grid Architecture for Advanced Network Applications

The research proposed here (shaded boxes) complements other proposals (indicated in purple italic type) submitted by colleagues (indicated in [bracketed] blue type). These proposals have been developed with the collective goal of defining and implementing an Integrated Grid Architecture. The EMERGE testbed ties together DoE university collaborators on the MREN GigaPoP by providing them with DiffServ routers, resource-specific software, and application tools, thereby enabling them to architect a seamless interoperable QoS problem-solving environment.